

mom

Please type a plus sign (+) inside this box → ☒PTO/SB/05 (1/98)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.**UTILITY
PATENT APPLICATION
TRANSMITTAL**

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No.

X-607 US

First Named Inventor or Appl. Identifier

David B. Parlour

Title

Intellectual Property Protection in a Programmable Logic Device

Express Mail Label No.

EL539649563US

APPLICATION ELEMENTS

See MPEP chapter 600 concerning utility patent application contents.

ADDRESS TO:

Assistant Commissioner for Patents
Box Patent Application
Washington, DC 20231

- 1.
- ☒
- Fee Transmittal Form (e.g., PTO/SB/17)
-
- (Submit an original, and a duplicate for fee processing)

- 6.
- ☐
- Microfiche Computer Program (Appendix)

- 2.
- ☒
- Specification [Total Pages
- 36**
-]
-
- (preferred arrangement set forth below)
-
- Descriptive title of the invention
-
- Cross References to Related Applications
-
- Statement Regarding Fed sponsored R & D
-
- Reference to Microfiche Appendix
-
- Background of the invention
-
- Brief Summary of the invention
-
- Brief Description of the Drawings (if filed)
-
- Detailed Description
-
- Claim(s)

- 7.
- ☐
- Nucleotide and/or Amino Acid Sequence Submission
-
- (if applicable, all necessary)

- 3.
- ☒
- Drawing(s) (35 USC 113) [Total Sheets
- 5**
-]
-
4. Oath or Declaration [Total Pages
- ☐

- a.
- ☐
- Computer Readable Copy

- b.
- ☐
- Paper Copy (identical to computer copy)

- c.
- ☐
- Statement verifying identity of above copies

- a.
- ☒
- Newly executed (original or copy)

- b.
- ☐
- Copy from a prior application (37 CFR 1.63(d))
-
- (for continuation/divisional with Box 17 completed)

- i.
- ☐
- [Note Box 5 below]
-
- DELETION OF INVENTOR(S)**
-
- Signed statement attached deleting
-
- inventor(s) named in the prior application,
-
- see 37 CFR 1.63(d)(2) and 1.33(b).

- 5.
- ☐
- Incorporation By Reference (useable if Box 4b is checked)
-
- The entire disclosure of the prior application, from which a
-
- copy of the oath or declaration is supplied under Box 4b,
-
- is considered as being part of the disclosure of the
-
- accompanying application and is hereby incorporated by
-
- reference therein.

ACCOMPANYING APPLICATION PARTS

- 8.
- ☒
- Assignment Papers (cover sheet & document(s))
-
- 9.
- ☐
- 37 CFR 3.73(b) Statement (when there is an assignee)
- ☐
- Power of Attorney
-
- 10.
- ☐
- English Translation Document (if applicable)
-
- 11.
- ☒
- Information Disclosure Statement (IDS)/PTO-1449
- ☒
- Copies of IDS Citations
-
- 12.
- ☐
- Preliminary Amendment
-
- 13.
- ☒
- Return Receipt Postcard (MPEP 503)
-
- (Should be specifically itemized)
-
- 14.
- ☐
- *Small Entity Statement filed in prior application, Status still proper and desired (PTO/SB/09-12)
-
- 15.
- ☐
- Certified Copy of Priority Document(s) (if foreign priority is claimed)
-
- 16.
- ☐
- Other:

* A new statement is required to be entitled to pay small entity fees, except where one has been filed in a prior application and is being relied upon.

17. If a
- CONTINUING APPLICATION**
- , check appropriate box and supply the requisite information below and in a preliminary amendment:

☐ Continuation☐ Divisional☐ Continuation-in-part (CIP)

of prior application No: _____

Prior application information: Examiner _____

Group / Art Unit: _____

18. CORRESPONDENCE ADDRESS☐ Customer Number or Bar Code Labelor ☒ Correspondence address below

(Insert Customer No. or Attach bar code label here)

Name

Edel M. Young

XILINX, INC.

Address

2100 Logic Drive

City

San Jose

State

California

Zip Code

95124

Country

US

Telephone

408-879-4969

Fax

408-377-6137

Name (print/Type)

Edel M. Young

Registration No. (Attorney/Agent)

32,451

Signature

Edel M. Young

Date

March 14, 2000

Burden Hour Statement. This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Assistant Commissioner for Patents, Box Patent Application, Washington, DC 20231.

INTELLECTUAL PROPERTY PROTECTION
IN A PROGRAMMABLE LOGIC DEVICE

David B. Parlour
Richard S. Ballantyne

TECHNICAL FIELD

This invention relates to a method and apparatus for securing configuration data used to configure a programmable logic device.

BACKGROUND INFORMATION

The term Programmable Logic Device (PLD) designates a class of devices that are programmable by end users to realize user-specific circuits. Examples of PLDs are FPGAs (Field Programmable Gate Arrays) and EPLDs (Erasable Programmable Logic Devices). To use a PLD, a user captures a circuit design using any of several capture/design tools. The user then uses the capture/design tool to convert the captured design into device-specific configuration data. The configuration data is commonly stored in an external storage device, such as an EPROM. Upon startup, the storage device supplies the configuration data to the PLD, thereby configuring the PLD to realize the user-specific circuit. Since the configuration data is often supplied in serial fashion, the configuration data is called a "bitstream". The PLD, having read in the configuration data bitstream, is then configured to realize the user-specific circuit.

Figure 1 (Prior Art) illustrates the loading of such a configuration data bitstream 1 from an external storage device 2 into a PLD 3 to realize a user-specific circuit.

1 PLD 3 in this case is an FPGA (for example, a XC4000 series
2 FPGA available from Xilinx, Inc. of San Jose, California).
3 PLD 3 includes a plurality of configurable logic blocks
4 (called CLBs) 4, and a programmable interconnect structure
5 5, surrounded by a ring of configurable input/output blocks
6 (called IOBs) 6. Each of the CLBs, the programmable
7 interconnect structure, and the IOBs includes configuration
8 memory cells, the contents of which determine how the CLBs,
9 the programmable interconnect structure, and the IOBs are
10 configured. Particular bits in bitstream 1 correspond to
11 the contents of particular configuration memory cells. If,
12 for example, two pieces of interconnect in the programmable
13 interconnect structure controlled by a particular memory
14 cell are to be connected in the user-specific circuit, then
15 the particular bit in the bitstream corresponding to the
16 memory cell for the particular programmable connection is
17 set accordingly. Upon power-up of the FPGA, the bitstream
18 1 is transferred from external storage device 2 into PLD 3
19 to configure PLD 3 to be the user-specific circuit. In
20 some prior art FPGA architectures, the protocol of the
21 configuration data bitstream (including knowledge of which
22 bits correspond to which configuration memory cells) is
23 proprietary to the FPGA manufacturer, thereby providing
24 individual users a level of security for their designs.
25 Without knowledge of the protocol and the significance of
26 the individual bits of the bitstream, another user cannot
27 readily regenerate the actual circuit by inspection of the
28 bitstream.

29 Over recent years, such user-specific circuits have
30 typically increased in size and complexity. Simultaneously,
31 market forces have reduced the amount of time practically
32 available for developing such large user-specific circuits.

1 In this environment, users have increasingly found it cost-
2 effective to purchase from IP vendors (intellectual
3 property vendors) pre-designed building blocks for use in
4 the users' designs. Such a building block, sometimes
5 called an "IP module" (intellectual property module), may
6 have taken the IP vendor several engineer-years to design.
7 An example of such an IP module is a PCI bus interface
8 module. Rather than taking the time to design a circuit to
9 perform the PCI interface function carried out by the IP
10 module, the user 7 purchases the IP module 8 (in digital
11 form) from the IP vendor. User 7 loads the IP module 8
12 into a capture/design tool 9 used to capture the user-
13 specific design. User 7 then adds other user-specific
14 circuitry 10 around the IP module using the capture/design
15 tool 9, thereby designing the overall user-specific
16 circuit. Once the overall user-specific circuit is
17 designed, simulated, placed and routed, (steps in
18 converting the user's design to device-specific
19 configuration data) the capture/design tool 9 outputs the
20 bitstream 1 for the overall user-specific circuit. As
21 illustrated in Figure 1, this bitstream 1 is then loaded
22 into the external storage device 2 (for example, a PROM) so
23 that the external storage device 2 will be able to supply
24 the bitstream 1 to the FPGA on power-up.

25 A problem, however, exists in that the user's user-
26 specific design can be copied. An unscrupulous second user
27 could obtain a product of a first user on the market and
28 copy the bitstream 1 that passes from the external storage
29 device 2 to FPGA 3 on power-up. The second user could then
30 use the copied bitstream to configure another FPGA (the
31 same type of FPGA used by the first user), thereby
32 replicating the first user's user-specific design and

1 product. Protection against this copying of one user's
2 design by another user is desired.

3 Erickson in U.S. Patent No. 5,970,142 [docket X-245]
4 discloses one method wherein the bitstream transferred from
5 the external storage device is in encrypted form and the
6 PLD being configured has a key to decrypt the encrypted
7 bitstream. The PLD receives the encrypted bitstream and
8 uses its key to generate the unencrypted bitstream which is
9 then loaded into the configuration memory cells to
10 configure the PLD. Because in this method the key is not
11 passed from the external storage device to the PLD, a
12 copier would not have access to the key. Without the key,
13 the copier would have a difficult time recovering the
14 bitstream. Other methods are also known.

15 Not only is one user's copying of another user's
16 design a problem, but the unauthorized reincorporation of a
17 vendor-supplied IP module into other user designs is also a
18 problem. Redman et al. in U.S. Patent No. 5,978,476, as
19 the present inventors understand it, discloses a design
20 processing system that attempts to verify the identity of
21 the user before allowing the user to use a vendor-supplied
22 IP module. The design processing system that generates a
23 programming file of bitstream information contains the IP
24 module in an encrypted form as well as a permission
25 verification system. The vendor to the IP module supplies
26 an authorization code to a particular user where the
27 authorization code is specific to the computer of the user
28 (or is specific to a "dongle" supplied to the user). When
29 an attempt is later made to use an IP module in the design
30 processing system, the permission verification system
31 requires the user to supply the authorization code. The
32 permission verification system reads the computer's

1 identification number (or the "dongle" number of a dongle
2 attached to the computer) and checks this number with the
3 supplied authorization code. If the number read from the
4 computer is not appropriate for the authorization number
5 provided by the user, then the user-verification process
6 fails and the permission verification system does not allow
7 the IP module to be decrypted. Aspects of the IP module
8 are not revealed to the user. Moreover, the design
9 processing system will not include configuration data for
10 the IP module in the output programming file.

11 If, on the other hand, the number read from the
12 computer is appropriate for the authorization code provided
13 by the user, then the permission verification system allows
14 the encrypted IP module to be decrypted and used by the
15 design processing system. The user uses the design
16 processing system to incorporate the IP module into the
17 user-specific circuit designed by the user. When design of
18 the user-specific circuit is completed, the design
19 processing system outputs configuration data for the user-
20 specific circuit in a programming file. The programming
21 file of configuration data is then usable to program a PLD
22 to realize the user-specific circuit. In this scheme,
23 however, the configuration data so generated is output from
24 the design processing system in unencrypted form. An
25 authorized but nonetheless unscrupulous user could copy the
26 programming file of bitstream information or a portion
27 thereof and reuse it in an unauthorized fashion outside the
28 control of the design processing system.

29 An improved system and method for protecting PLD
30 designs is desired wherein a user is prevented from using
31 an IP module in an unauthorized manner, and wherein one

1 user is prevented from copying the user-specific circuit of
2 another user.

3
4 SUMMARY

5 A user arranges with an IP vendor to use a particular
6 IP module. If the arrangement is suitable to the IP
7 vendor, then the IP vendor issues the user an authorization
8 code. This authorization code contains: 1) a usage
9 condition, 2) an indication of the IP module authorized, 3)
10 an "IP module key" for the IP module, and 4) a value (for
11 example, a serial number or ID number or dongle number)
12 that identifies the user's development system. There are
13 numerous different usage conditions. Possible usage
14 conditions include: usage is authorized for an unlimited
15 number of uses, usage is authorized for an unlimited number
16 of uses during a particular time period, usage is
17 authorized for a limited number of uses, usage is
18 authorized only for a particular development system (node-
19 locked license), usage is authorized on a particular
20 individual PLD.

21 In an embodiment where authorization is granted for
22 use on a particular PLD, the authorization code also
23 contains the unique device identifier (UDI) of the target
24 PLD to be programmed. For some types of PLD, each PLD is
25 manufactured with its own unique, non-volatile, non-
26 rewritable UDI that uniquely identifies that particular
27 PLD. The user uses the development system to read the UDI
28 out of the target PLD. The user then supplies the UDI to
29 the IP vendor. The IP vendor uses authorization code
30 generating software to generate an authorization code that
31 includes that UDI. The authorization code is therefore, in
32 this embodiment, specific to the particular target PLD.

1 A public key/private key encryption scheme is used to
2 encrypt the authorization code such that the user cannot
3 decipher the information contained within it. The IP
4 vendor's authorization code generating software uses the
5 UDI (as a public key) as well as another key (a private
6 key) to encrypt the authorization code. The private key is
7 known both to the IP vendor's authorization code generating
8 software as well as to the user's development system, but
9 the private key (unlike the UDI) is not communicated to the
10 authorization code generating software. The private key is
11 not known to the user. The user and the IP vendor
12 therefore have no access to the private key.

13 After receiving the authorization code, a part of the
14 development system called the license manager "verifies"
15 that the user is authorized to use the particular IP
16 module. If, for example, the usage condition indicated by
17 the authorization code indicates use is authorized during a
18 certain time period, then the license manager consults a
19 clock maintained by the license manager. If the current
20 time as read from the clock is within the authorized time
21 period, then the license manager verifies usage of the IP
22 module. Alternatively, if the usage condition indicates
23 use is authorized for a given number of usages, then the
24 license manager consults a usage meter maintained by the
25 license manager. If the usage meter indicates that the
26 authorized number of uses has not been exceeded, then the
27 license manager increments the usage meter and verifies
28 usage of the IP module. Alternatively, if the usage
29 condition indicates use is authorized on one or more
30 particular PLDs, then the license manager reads the UDI out
31 of the target PLD, decrypts the UDI portion of the
32 authorization code, and verifies that the decrypted UDI

1 matches the UDI read from the PLD. If the decrypted UDI
2 matches the UDI read from the PLD, then the license manager
3 verifies usage of the IP module on the target PLD.

4 If the verification is made, then the license manager
5 uses the UDI (as a public key) along with the private key
6 to encrypt the "IP module key" from the authorization code.
7 Once encrypted, the license manager sends the encrypted IP
8 module key to the target PLD. The target PLD receives the
9 encrypted IP module key and decrypts it using two keys
10 stored on the target PLD: 1) the UDI (public key), and 2)
11 the private key. The private key, unlike the UDI, cannot
12 be read out of the PLD. The resulting decrypted IP module
13 key is then stored in non-volatile memory on the PLD in
14 association with a key number. The target PLD can use the
15 key number to look up the key at a later time. If,
16 however, the verification is not made, then the encrypted
17 key is not sent to the target PLD.

18 When the user has completed the design of the user-
19 specific circuit that incorporates the IP module, a
20 configuration data bitstream is to be generated so that it
21 can be sent to the target PLD to configure the target PLD.
22 A capture/design tool in the development system outputs
23 this configuration bitstream, but before it is sent to the
24 target PLD, the license manager encrypts the IP module
25 portion of the bitstream using the IP module key. The
26 license manager also inserts into the bitstream the key
27 number of the IP module key in such a way that the key
28 number is associated in the bitstream with the encrypted IP
29 module portion.

30 The target PLD receives the bitstream including the
31 key number and the encrypted IP module portion. The target
32 PLD uses the key number to retrieve the proper IP module

1 key from its non-volatile memory, and then uses the
2 retrieved IP module key to decrypt the IP module portion.
3 When the IP module portion has been decrypted, the
4 resulting configuration data bitstream is used to configure
5 the target PLD so as to realize the user-specific circuit.

6 Using this approach, individual IP vendors can
7 directly license their IP modules to PLD users without
8 involvement of the PLD manufacturer. Numerous IP modules
9 can be used in a single user-specific design, the user
10 having received a different authorization code for the use
11 of each IP module. Because the IP module keys used to
12 decrypt the IP module portions of the bitstream are
13 themselves encrypted in a way specific to the particular
14 target PLD (due to use of the UDI), IP vendors can license
15 use of their IP modules for particular PLDs or for a given
16 number of uses ("use-based licensing"). IP vendors can
17 also license their IP modules for a time period or up until
18 a particular expiration date ("time-based licensing").

19 Not only are IP module keys for IP modules encrypted
20 and sent to the target PLD, but a "user key" for the
21 portion of the user-specific design contributed by the user
22 is also encrypted and sent to the target PLD. No
23 authorization code is required for the encryption of the
24 user key and the passing of this user key to the PLD. The
25 use of this user key to decrypt the appropriate portion of
26 the bitstream occurs as described above in connection with
27 the IP module keys. In this way, the user-specific design
28 of a first user is protected from copying by a second user
29 because the portion of the user-specific circuit designed
30 by the first user is encrypted with a user key not known to
31 the second user. The IP modules of the IP vendors are
32 secure from copying by a user because the IP module

1 portions of the bitstream are encrypted with IP module keys
2 not known to the user.

3 Other structures and methods are disclosed in the
4 detailed description below. This summary does not purport
5 to define the invention. The invention is defined by the
6 claims.

7 8 BRIEF DESCRIPTION OF THE DRAWINGS

9 Figure 1 (Prior Art) is a block diagram illustrating
10 conventional loading of a configuration bitstream into a
11 programmable logic device (PLD).

12 Figure 2 is a block diagram of a system wherein a
13 configuration bitstream is loaded into a programmable logic
14 device (PLD) in accordance with an embodiment of the
15 present invention.

16 Figure 3 is a simplified top-down diagram of a central
17 part of the PLD of Figure 2 configured to involve three IP
18 modules.

19 Figure 4 is a simplified flow-chart of a method
20 carried out by the system 100 of Figure 2.

21 Figure 5 is a simplified diagram illustrating the
22 intellectual property protection circuit 112 of the PLD of
23 Figure 2.

24 Figure 6 is a simplified diagram of the configuration
25 bitstream that configures the PLD of Figure 2.

26 Figure 7 is a flowchart of another method carried out
27 by the system 100 of Figure 2.

28 29 DETAILED DESCRIPTION

30 Figure 2 is a simplified diagram illustrating a system
31 100 that carries out a method in accordance with an
32 embodiment of the present invention. A configuration

1 bitstream 101 for a user-specific circuit is loaded into a
2 target FPGA integrated circuit 102 in such a way that
3 vendor-supplied IP module design information in the
4 bitstream 101 is protected from unauthorized use. System
5 100 includes authorization code generating software 103, a
6 development system 104, an external storage device 105, and
7 target FPGA 102. The development system 104 typically
8 involves software executing on a personal computer or a
9 workstation and includes a capture/design tool 106 and a
10 license manager 107. A user 108 uses the capture/design
11 tool 106 to design the user-specific circuit, to simulate
12 it, to map, place and route it on FPGA 102, and to generate
13 the configuration bitstream 101 for configuring FPGA 102.
14 For additional information on an exemplary development
15 system, see: pages 2-1 through 2-11 of "The Programmable
16 Logic Data Book 1999", copyright 1999 by Xilinx, Inc. (the
17 content of which is incorporated herein by reference).

18 FPGA 102 includes a two-dimensional array of
19 configurable logic blocks (CLBs) 4 and a programmable
20 interconnect structure 5 surrounded by a ring of
21 programmable input/output blocks (IOBs) 6. An intellectual
22 property protection circuit (IPPC) 112 is included in a
23 corner of the FPGA 102. FPGA 102 may, for example, be a
24 Virtex family FPGA available from Xilinx, Inc. of 2100
25 Logic Drive, San Jose, California 95124. For additional
26 information on a Virtex family FPGA, including information
27 on the structure and operation of a CLB, an IOB and a
28 programmable interconnect structure, see: pages 3-3 through
29 3-22 of "The Programmable Logic Data Book 1999", copyright
30 1999 by Xilinx, Inc. (the content of which is incorporated
31 herein by reference).

1 Figure 3 is a simplified top-down diagram of the user-
2 specific circuit as it is realized in the configured FPGA
3 102. The square 300 of Figure 3 represents the central CLB
4 and programmable interconnect area 4 and 5 of FPGA 102.
5 The user-specific circuit includes three IP modules 301-
6 303. In this example, a different piece of IP module
7 design information is used to realize each of these modules
8 and each piece is provided by a different IP vendor. User
9 108 (see Fig. 2) has incorporated these IP modules 301-303
10 into the overall user-specific circuit by providing
11 additional circuitry 304. User 108 designs this additional
12 circuitry 304 using capture/design tool 106. In the
13 illustrated example, the circuitry of each of the three
14 modules 301-303 is placed and routed in its own separate
15 area of FPGA 102.

16 Figure 4 (comprising Figures 4A and 4B) is a flowchart
17 of a method carried out by the system 100 of Figure 2.
18 Generally, the user 108 seeks to incorporate an IP module
19 (for example, IP module 301) of an IP vendor (for example,
20 IP vendor 113) into the user's user-specific circuit. The
21 user 108 communicates 114 with the IP vendor 113 and if a
22 license arrangement satisfactory to IP vendor 113 is made,
23 then IP vendor 113 supplies the user 108 with an
24 authorization code 115 to use the IP module. The
25 authorization code includes: 1) a usage condition, 2) an
26 indication of the particular IP module authorized, 3) an
27 "IP module key" for the IP module, and 4) a value (for
28 example, a serial number or ID number or dongle number)
29 that identifies the user's development system. There are
30 numerous different usage conditions that can be specified.
31 Possible usage conditions include: usage is authorized for
32 an unlimited number of uses, usage is authorized for an

1 unlimited number of uses during a particular time period,
2 usage is authorized for a limited number of uses, usage is
3 authorized on particular individual PLDs.

4 In the method of Figure 4, the authorization code 115
5 grants authorization to use one particular development
6 system to one specified IP module on one particular target
7 PLD only. In such a case where the authorization code
8 grants authorization for one particular PLD, the
9 authorization code also contains a unique device identifier
10 (UDI) 116 of the target PLD to be programmed. Each FPGA
11 carries its own unique UDI that is programmed into it by
12 the FPGA manufacturer. Once a UDI is written into a FPGA
13 at the end of FPGA manufacturing, the UDI cannot be
14 rewritten. A UDI therefore uniquely identifies a
15 particular FPGA.

16 Figure 5 illustrates the UDI 116 stored in a write-
17 once, non-volatile, read-only-memory (ROM) location in
18 intellectual property protection circuit 112. This non-
19 volatile UDI storage can be implemented in any technology
20 that can be added to the FPGA. The non-volatile UDI
21 storage can, for example, be implemented using an antifuse-
22 based register, a fuse-based register, a laser-programmed
23 register, an EPROM register, and/or a flash-based register.

24 IP vendor 113 queries user 108 for the UDI 114 of the
25 target FPGA 102. In one embodiment, the user 108 uses the
26 development system 104 to read the UDI from the target FPGA
27 102. In this embodiment, development system 104 includes
28 associated interface hardware (not shown) and this
29 interface hardware reads the UDI 116 out of FPGA 102. The
30 interface hardware is provided with the development system
31 so that the development system 104 can read from and/or
32 write to FPGA 102. The user may, however, obtain the UDI

1 116 by means other than such interface hardware. The UDI
2 may, for example, simply be printed on the FPGA 102.

3 When the IP vendor 113 receives UDI 116, the IP vendor
4 113 uses authorization code software 103 to generate the
5 authorization code 115 such that the information contained
6 in the authorization code is encrypted using a public
7 key/private key scheme. The public key used is UDI 116
8 received from user 108. The private key used is a private
9 key 117 known to the authorization code generating software
10 103, to the license manager 107, and to PLD 102. The
11 private key 117, unlike UDI 116, does not pass through user
12 108 and is therefore not known to user 108. The private
13 key 117 cannot be read out of the authorization code
14 generating software 103, out of the development system 104,
15 or out of PLD 102.

16 Once the authorization code 115 is generated, IP
17 vendor 113 sends (Fig. 4, step 200) the authorization code
18 115 to the user 108. When user 108 attempts to use IP
19 module design information 118 (in this example, IP module
20 design information 118 is design information for realizing
21 IP module 301), the license manager 107 queries (step 201)
22 user 108 for authorization code 115. User 108 responds
23 (step 202) by supplying the authorization code 115 to the
24 license manager 107. The license manager 107 uses the UDI
25 116 (the public key) and the private key 117 to decrypt the
26 authorization code 115 so as to recover: 1) the usage
27 condition, 2) the indication of the IP module authorized,
28 3) the "IP module key", and 4) the serial number or ID
29 number or dongle number of the user's development system.

30 Next, license manager 107 reads (step 203) the UDI 116
31 from the FPGA 102 and reads the serial number or ID number
32 or dongle number of the development system 104. The

1 license manager 107 then verifies (step 204) that the
2 authorization code 115 supplied by user 108 in fact grants
3 access to the IP module design information 118. The
4 license manager 107 does this by: 1) checking that the UDI
5 116 read from FPGA 102 matches the UDI from the
6 authorization code 115, 2) checking that the serial number
7 or ID number or dongle number read from the user's
8 development system 104 matches the serial number or ID
9 number or dongle number from authorization code 115, and 3)
10 checking that the usage condition of authorization code 115
11 is satisfied.

12 In this embodiment, if any of these checks fails, then
13 license manager 107 does not verify authorization to use
14 the IP module. The license manager 107 denies the user 108
15 access to the IP module design information 118. In one
16 example, IP module design information 118 has been loaded
17 into the capture/design tool 106 but is present in
18 encrypted form. The IP module design information 118 may,
19 for example, have been publicly available on the World Wide
20 Web in its encrypted form and may have been downloaded by
21 user 108 and loaded into development system 104 for
22 intended future use. When authorization to use the IP
23 module is not verified, the license manager 107 does not
24 decrypt the IP module design information 118. The
25 capture/design tool 106 is therefore unable to use the IP
26 module design information 118 and the user 108 is unable to
27 use the IP module design information 118 in the user-
28 specific circuit.

29 If, on the other hand, the license manager 107
30 succeeds in checking all these conditions, then license
31 manager 107 verifies authorization to use the IP module and
32 allows (step 205) user 108 access to the IP module design

1 information 118 via the capture/design tool 106. License
2 manager 107 uses the IP module key 120 from the
3 authorization code 115 and an encryptor/decryptor 119 to
4 decrypt the encrypted IP module design information 118.
5 Once decrypted, the IP module design information 118 is
6 accessible to user 108 via capture/design tool 106. User
7 108 can then design the IP module 301 into the user-
8 specific circuit.

9 The IP module key 120 has a key number. In the
10 example of Figure 2, key 120 associated with IP module
11 design information 118 and IP module 301 is key number one.
12 Other such keys 121-123 associated with IP modules 302-303
13 and user's circuitry 304 are also shown in Figure 2 and are
14 key numbers two, three and four, respectively. The keys
15 121-122, like key 120, are keys for IP modules and
16 therefore would have been received by license manager 107
17 via authorization codes. Key 123, on the other hand, is a
18 key for user's circuitry 304. Key 123 is not received via
19 an authorization code but rather is supplied to the license
20 manager 107 by user 108.

21 In addition to decrypting the IP module design
22 information 118, license manager 107 retrieves (step 205)
23 the key 120 for IP module design information 118, encrypts
24 key 120 and its key number using the UDI 116 (as a public
25 key) and the private key 117, and then sends to FPGA 102 a
26 key bitstream 125 involving the encrypted key 120 and its
27 key number. This key bitstream 125 is received in FPGA
28 102, in one embodiment, via a JTAG boundary scan port
29 terminal.

30 FPGA 102 uses its UDI 116 (a public key), private key
31 117, and an encryptor/decryptor 124 to decrypt (step 206)
32 the encrypted key and key number. The FPGA 102 then stores

1 the key 120 in association with the key number. In the
2 illustrated example, key 120 is key number one and is
3 written into the first location of a one-time writable ROM
4 that holds UDI 116 and private key 117. Unlike UDI 116, IP
5 module keys 120-123 and private key 117 are not readable
6 from outside FPGA 102 but rather are only readable by
7 encryptor/decryptor 124. Private key 117 is written into
8 the FPGA by the FPGA manufacturer at the end of FPGA
9 manufacturing. Private key 117 is not known to the IP
10 vendor 113 nor to user 108 and never passes between FPGA
11 102 and development system 104. The same private key 117
12 is used by: 1) all FPGAs manufactured by a particular FPGA
13 manufacturer, 2) the development systems used to program
14 those FPGAs, and 3) the authorization code generating
15 software that is used to supply authorization codes to the
16 development systems.

17 Similar steps are carried out to obtain authorization
18 codes and keys for the other IP modules 302 and 303 of the
19 user-specific circuit. If IP module design information for
20 another IP module is provided by a different IP vendor,
21 then user 108 arranges with that IP vendor to receive a
22 proper authorization code to access that particular IP
23 module. Accordingly, each IP vendor can apply its own
24 licensing restrictions on its own IP module design
25 information. In the example of Figure 2, keys 121 and 122
26 are the keys for IP modules 302 and 303 (Figure 3),
27 respectively.

28 When the user 108 has completed the design of the
29 user-specific circuit employing IP module design
30 information for IP modules 301-303, the capture/design tool
31 106 outputs a bitstream (step 207) for the composite user-
32 specific circuit. Before the portion of the bitstream

1 carrying the configuration data for a particular IP module,
2 the license manager 107 inserts a start code and then the
3 key number of the key for that IP module. The license
4 manager 107 uses encryptor/decryptor 119 to encrypt the
5 associated configuration data using the key indicated by
6 the preceding key number.

7 Figure 6 is a simplified diagram of bitstream 101.
8 Note that the portion of the bitstream associated with IP
9 module 301 involves a start code 305, the key number for
10 key 120, and configuration data that is encrypted with key
11 120. In the example of Figure 6, each of the portions of
12 the bitstream for IP modules 301-303 is a contiguous block.
13 This need not be the case. Note that the configuration
14 data for the user's portion of the circuit 304 is split
15 into three parts. Each of the parts has a start code, the
16 key number of the user key (key 123) used to encrypt the
17 configuration data, and a part of the configuration data.
18 Splitting up of the configuration data in this way is
19 facilitated by the fact that the configuration data
20 includes address information identifying memory cells on
21 FPGA 102 as well as data that is to be written into those
22 identified memory cells. The bit stream 101 is sent (step
23 208) to a serial data input (DIN) terminal of FPGA 102.

24 Next, FPGA 102 receives the bitstream 101 via the
25 FPGA's DIN terminal and then uses encryptor/decryptor 124
26 to decrypt (step 209) each encrypted part with its
27 respective key. For example, when FPGA 102 receives the
28 part of the bitstream 101 corresponding to IP module 301,
29 it receives the start code 305 and then the key number for
30 key 120. This key number is "one". FPGA 102 uses this key
31 number "one" to retrieve the key 120 stored in association
32 with key number "one" in the one-time writable non-volatile

1 ROM. Key 120 is supplied to encryptor/decryptor 124 to
2 decrypt the following configuration data for IP module 301.
3 In this way, the encrypted configuration data for each IP
4 module is decrypted using the correct key identified by the
5 preceding key number in the bitstream. The configuration
6 data from the resulting decrypted bitstream 126 is loaded
7 (step 209) into appropriate memory cells of the CLBs 109,
8 IOBs 111, and configurable interconnect structure 110 so as
9 to configure FPGA 102 to realize the user-specific circuit.
10 Because FPGA 102 in this example is a Virtex family Xilinx
11 FPGA, the resulting decrypted bitstream 126 comports with
12 the standard Xilinx bitstream protocol for configuring a
13 Virtex family FPGA. If, on the other hand, the PLD being
14 configured is not a Xilinx Virtex FPGA and therefore
15 requires a different configuration bitstream, then
16 decrypted bitstream 126 would comport with the protocol
17 required by that PLD.

18 Various encryption methods and structures are known
19 and available for use in implementing system 100. In one
20 embodiment, encryptor/decryptor 119 and encryptor/decryptor
21 124 are DES (Data Encryption Standard)
22 encryptor/decryptors. Encryptor/decryptor 119 is, for
23 example, realized in software whereas encryptor/decryptor
24 124 is realized in hardware. The DES algorithm has been a
25 nationwide standard since about 1976. For additional
26 information and implementation considerations involved in
27 realizing DES encryptors and decryptors in software and
28 hardware, see: 1) Applied Cryptography, second edition, by
29 Bruce Schneier, Chapter 12: Data Encryption Standard (DES),
30 pages 265-301 (1996); 2) U.S. Patent Numbers 5,671,284 and
31 5,835,599 to Buer; and 3) U.S. Patent Number 4,731,843 to

1 Holmquist (the content of these documents is incorporated
2 herein by reference).

3 It is desired that one user not be able to understand
4 or copy the design of another user by copying the bitstream
5 loaded into the FPGA on power-up. In the above-described
6 embodiment, the user provides key "KEY 4" 123 and this key
7 is used to encrypt the portions of the bitstream associated
8 with user's portion 304 of the user-specific circuit. A
9 second user cannot therefore decipher the first user's
10 design by examining the bitstream. The second user cannot
11 copy the bitstream and program other FPGAs, even if the
12 second user were to arrange with the necessary IP vendors
13 to use the needed IP modules, because the second user would
14 not know "KEY 4" of the first user. Because each FPGA
15 programmed must first be loaded with the set of keys 119
16 from the license manager, and because the bitstream by
17 which these keys is loaded is not the same from FPGA to
18 FPGA, copying the bitstream by which the keys are loaded
19 into one FPGA would not enable the second user to load the
20 keys into another FPGA.

21 It is also desired that a user not be able to
22 understand or copy the IP module design from an IP vendor.
23 In the above-described embodiment, the portion of the
24 configuration bitstream 101 corresponding to an IP module
25 is encrypted with a key that is never known by the user.
26 The user cannot therefore decrypt the IP module. The user
27 cannot steal the key by monitoring the transfer of keys to
28 the FPGA because the keys are encrypted using the UDI (a
29 public key) and a private key not known to the user.

30 It is also desired that the configuration bitstream
31 101 itself (as opposed to the bitstream 125 by which the
32 keys are loaded) be identical for all FPGAs of a given

1 user-specific design. This facilitates rapid programming
2 of the user-specific circuit design into multiple FPGAs at
3 the same time. In the above-described embodiment,
4 bitstream 101 is the same for all FPGAs programmed with the
5 same user-specific circuit. Key bitstream 125 by which the
6 keys are loaded changes from FPGA to FPGA, but the
7 configuration data bitstream 101 remains the same.

8 Care is also taken to ensure that a user cannot
9 arrange with an IP vendor to use an IP module for a limited
10 time or a limited number of usages, gain access to the IP
11 module in an authorized manner for example via
12 capture/design tool 106, but then examine the internal
13 design of the IP module via the capture/design tool, copy
14 the design, and use the copied design again outside the
15 control of the IP vendor.

16 In the capture/design tool 106, a representation of
17 the user-specific circuit complete with the IP modules 301-
18 303 exists in a netlist form. Ordinarily a user has been
19 able to view a net in any portion of the user-specific
20 circuit. The user could, for example, select a net and
21 then view in graphical form all circuit elements to which
22 it is connected. Doing this for all nets in a circuit
23 reveals the circuit. The user could ordinarily view any
24 net in the user-specific circuit at the logic netlist
25 stage, or after place and route and the logic-optimized
26 netlist stage. In accordance with one embodiment of the
27 present invention, each net of the user-specific circuit
28 carries a "visible/invisible" attribute. All nets inside
29 an IP module carry the "invisible" attribute. If the user
30 were to attempt to use the capture/design tool to view a
31 net inside an IP module in an attempt to redraw the
32 circuitry, the capture/design tool 106 would detect the

1 "invisible" attribute of the net and not allow the user to
2 examine the net. Nets outside the IP modules, on the other
3 hand, carry the "visible" attribute and are visible to the
4 user via the capture/design tool 106. The "invisible"
5 attribute on a net would also prevent the user from viewing
6 signals on the net during simulation.

7 Because the keys are written into each respective FPGA
8 using a unique UDI that is first read from the particular
9 FPGA being programmed, the license manager 107 is able to
10 control which particular FPGAs are programmed. This, in
11 combination with an authorization code that is specific to
12 a particular target FPGA, allows an individual IP vendor to
13 employ "use-based" licensing of an IP module.

14 An IP vendor may also employ "time-based" licensing of
15 an IP module. In one embodiment, IP vendor 113 gives user
16 108 an authorization code 115 to use an IP module for a
17 certain period of time or up until a certain expiration
18 date. IP vendor 113 loads the time period or expiration
19 date into license manager 107. Once the licensing manager
20 determines (for example, using a clock internal to the
21 license manager) that the authorized time period has
22 expired or the expiration data has passed, the licensing
23 manager 107 no longer verifies that authorization 115
24 authorizes usage of the IP module.

25 It is therefore seen that a single FPGA can involve
26 multiple IP modules, where each IP module is licensed by a
27 different IP vendor using a different license and business
28 arrangement without the intervention of the FPGA
29 manufacturer. The direct interaction between IP vendors
30 and users is generally advantageous from the FPGA
31 manufacturer's point of view, because it eliminates the
32 need for the FPGA manufacturer to act as a royalty

1 collector on behalf of IP vendors. Moreover, the ability
2 of an end user to program FPGAs with selected IP modules
3 without being able to steal the IP module design eliminates
4 the need for the FPGA manufacturer or the IP vendor to
5 inventory FPGAs partially programmed with individual IP
6 modules.

7 In the example of Figure 4 the user cannot design an
8 IP module into the user-specific design before a suitable
9 license arrangement has been made because authorization
10 code 115 is required in order for the capture/design tool
11 106 to access the IP module design information 118. This
12 need not, however, be the case. In one embodiment, the
13 capture/design tool 106 decrypts the IP module design
14 information 118 and allows the user access to the IP module
15 design information 118 thereby allowing the user to design
16 the IP module into a user-specific design without any
17 authorization code. The user cannot view nets within the
18 IP module or view simulation signals inside the IP module.
19 The IP module therefore appears as a black box and is
20 fairly secure from being reverse-engineered by the user.
21 Only at the point that the user-specific design is ready
22 for downloading to FPGA 102 as bitstream 101 does the user
23 have to obtain the appropriate IP module licenses. When
24 the user attempts to create configuration bitstream 101,
25 the license manager reads the UDI 116 from FPGA 102 and
26 asks the user for authorization code 115. If, as in the
27 method of Figure 4, the user provides an appropriate
28 authorization code (that, among other things, contains the
29 UDI 116 read from target FPGA 102), then the license
30 manager 107 sends key bitstream 125 to the target FPGA 102
31 and allows the encrypted configuration bitstream 101 to be
32 generated.

1 Figure 7 is a flowchart of another method carried out
2 by system 100 of Figure 2. In this method, a different
3 configuration bitstream is generated for each different PLD
4 to be configured even if the same user-specific circuit is
5 being programmed into each PLD. Steps 400-404 of the
6 method of Figure 7 are the same as steps 200-204 of the
7 method of Figure 4. In step 405, the license manager 107
8 allows access to the IP module, but does not encrypt a key
9 and key number and does not send them to the PLD 102 being
10 programmed. PLD 102 therefore does not decrypt keys and
11 key numbers, nor does PLD 102 store any such decrypted
12 keys. When the user has designed the user-specific circuit
13 employing the IP module, a configuration bitstream is
14 output (Step 407) by capture/design tool 106. No key
15 numbers are inserted into the bitstream as in step 207 of
16 Figure 4, rather the entire configuration bitstream is
17 encrypted using a UDI-dependent key and private key 117.
18 The UDI-dependent key is, on one embodiment, UDI 116 as
19 read from PLD 102. UDI 116 is, however, also stored on PLD
20 102. PLD 102 uses the UDI-dependent key along with private
21 key 117 to decrypt (Step 408) the incoming encrypted
22 configuration bitstream 101 and then configures PLD 102
23 using the resulting bitstream so as to realize the user-
24 specific circuit. Because the incoming encrypted
25 configuration bitstream 101 is encrypted using a key that
26 depends on the UDI 116 of the particular target PLD 102,
27 target PLD 102 is the only PLD that can decrypt and use
28 this configuration bitstream.

29 FPGA 102 implements a read-back mechanism usable to
30 read out the bitstream and/or to examine internal states of
31 the FPGA. In one embodiment, a read back bitstream passes
32 out of FPGA 102 via encryptor/decryptor 124 (see Figure 5).

1 A readback key known to license manager 107 (for example, a
2 UDI-dependent key that is derived in a particular way from
3 UDI 116) is used to encrypt the read-back bitstream. This
4 readback key may be, for example, generated from UDI 116
5 via hardware and as such is not stored in non-volatile
6 memory on FPGA 102.

7 The license manager 107, having previously read UDI
8 116, also knows how the readback key is generated from UDI
9 116. The license manager 107 is therefore able to obtain
10 the readback key and to use the readback key along with
11 encryptor-decryptor 119 to decrypt the read-back bitstream.
12 License manager 107 controls capture/design tool 106 so
13 that the user can only examine those portions of the
14 readback data and states that do not reveal the inner
15 workings of secure IP modules. This gives the user access
16 to information about the portions of the user-specific
17 design that the user controls, while protecting the secure
18 IP modules from reverse-engineering attempts.

19 A commercially-available general purpose license
20 management system involving a license manager and
21 authorization code generating software may be adapted to
22 serve as license manager 107 and authorization code
23 generating software 103. For example, the Flexible License
24 Manager (FLEXlm) (also called the "Highland License
25 Manager") available from GLOBEtrotter Software, Inc. of
26 Cupertino, California may be employed. The UDI,
27 development system serial number, and/or other information
28 in the authorization code is not, in some embodiments,
29 encrypted. Rather, it is present in the authorization code
30 in unencrypted form but is combined with a specially
31 generated Message Authentication Code (MAC). The MAC is a
32 complex checksum produced from the rest of the

1 authorization code in a way not known to the user. Because
2 the user is not aware of how the checksum was produced, the
3 user is unable to generate his/her own valid authorization
4 codes.

5 Although the present invention is described in
6 connection with certain specific embodiments for
7 instructional purposes, the present invention is not
8 limited thereto. The disclosed embodiments are applicable
9 to one-time programmable PLDs and FPGAs (including
10 antifuse- or oxide-rupture-based, fuse-based, and laser-
11 programmed PLDs and FPGAs), to non-volatile PLDs and FPGAs
12 (including EPROM-based and flash-based PLDs and FPGAs), as
13 well as to RAM-based PLDs and FPGAs. The license manager
14 may read the UDI from the PLD to be programmed and then
15 receive the authorization code from the user, or
16 alternatively may read the UDI from the PLD to be
17 programmed after receiving the authorization code from the
18 user. In some embodiments, a user can use the
19 capture/design tools to design an IP module into the user-
20 specific circuit without having supplied a suitable
21 authorization code, the license manager only requiring a
22 proper authorization code if a final output configuration
23 bitstream is to be generated. IP module keys can be
24 transferred from the license manager to the target PLD
25 without the use of a public key/private key encryption
26 scheme. Other ways of protecting the IP module keys from
27 unauthorized use can be used. In one embodiment, the IP
28 module keys are encrypted just with the UDI and then are
29 transferred to the target PLD. The authorization code need
30 not contain a serial number or ID number or dongle number
31 identifying the user's development system. In some
32 embodiments, an authorization code grants access to an IP

1 module on any development system, provided that the usage
2 conditions indicated by the authorization code are met.
3 Accordingly, various modifications, adaptations, and
4 combinations of various features of the described
5 embodiments can be practiced without departing from the
6 scope of the invention as set forth in the claims.

2025 RELEASE UNDER E.O. 14176

1 CLAIMS:

2 What is claimed is:

3

4 1. A method, comprising:

5 (a) maintaining a device identifier and a private key
6 in a programmable logic device, the device identifier and
7 the private key being non-volatile such that if power to
8 the programmable logic device is lost the device
9 identifier and private key remain stored in the
10 programmable logic device;

11 (b) receiving a first encrypted key onto the
12 programmable logic device, and using the device
13 identifier and the private key to decrypt the first
14 encrypted key thereby generating a first key;

15 (c) receiving onto the programmable logic device a
16 bitstream comprising first encrypted configuration data
17 encrypted with the first key;

18 (d) using the first key to decrypt the first encrypted
19 configuration data on the programmable logic device
20 thereby generating first configuration data; and

21 (e) configuring a first portion of the programmable
22 logic device using the first configuration data.

23

24 2. The method of Claim 1, wherein neither the device
25 identifier nor the private key are rewritable.

26

27 3. The method of Claim 1, wherein the bitstream further
28 comprises a first key number associated with the first
29 encrypted configuration data, the first key being stored
30 on the programmable logic device in association with the
31 first key number, the programmable logic device in step
32 (d) using the first key number in the bitstream to

1 identify the first key as the key that will be used in
2 step (d) to decrypt the first encrypted configuration
3 data.

4

5 4. The method of Claim 1, wherein the device identifier
6 and the private key are stored on the programmable logic
7 device in one of the group consisting of: an antifuse-
8 based storage element, a fuse-based storage element, a
9 laser-programmed storage element, an EPROM storage
10 element, and a flash-based storage element.

11

12 5. The method of Claim 1, further comprising:
13 after the first key is generated in step (b), storing
14 the first key in non-volatile memory on the programmable
15 logic device.

16

17 6. The method of Claim 1, wherein the first encrypted
18 configuration data is decrypted in step (d) on the
19 programmable logic device by a hardware decryptor.

20

21 7. The method of Claim 1 wherein
22 step (b) further comprises receiving a second encrypted
23 key onto the programmable logic device and using the
24 device identifier and the private key to decrypt the
25 second encrypted key, thereby generating a second key;
26 step (c) further comprises receiving onto the
27 programmable logic device a bitstream comprising second
28 encrypted configuration data encrypted with the second
29 key;
30 step (d) further comprises using the second key to
31 decrypt the second encrypted configuration data on the

1 programmable logic device, thereby generating second
2 configuration data; and
3 step (e) further comprises configuring a second
4 portion of the programmable logic device using the second
5 configuration data.

6

7 8. The method of Claim 7, wherein the bitstream further
8 comprises a first key number associated with the first
9 encrypted configuration data, and wherein the bitstream
10 further comprises a second key number associated with the
11 second encrypted configuration data, the first key being
12 stored on the programmable logic device in association
13 with the first key number, the second key being stored on
14 the programmable logic device in association with the
15 second key number, the programmable logic device in (d)
16 using the first key number in the bitstream to identify
17 the first key as the key that will be used in (d) to
18 decrypt the first encrypted configuration data, the
19 programmable logic device in (d) using the second key
20 number in the bitstream to identify the second key as the
21 key that will be used in (d) to decrypt the second
22 configuration data.

23

24 9. The method of Claim 7 further comprising:
25 after the first key and the second key are generated in
26 step (b), storing the first key and the second key in
27 non-volatile memory on the programmable logic device.

28

29 10. The method of Claim 7, wherein the first portion of
30 the programmable logic device is configured in (e) to
31 realize a first IP module, and wherein the second portion

1 of the programmable logic device is configured in (e) to
2 realize a second IP module.

3

4 11. The method of Claim 1, wherein the programmable logic
5 device is an SRAM-based PLD.

6

7 12. The method of Claim 10, wherein the non-volatile
8 memory in the programmable logic device is flash-based.

9

10 13. The method of Claim 10, wherein the non-volatile
11 memory in the programmable logic device is one-time
12 programmable.

13

14 14. The method of Claim 10, wherein the non-volatile
15 memory in the programmable logic device is antifuse-
16 based.

17

18 15. The method of Claim 10, wherein the non-volatile
19 memory in the programmable logic device is fuse-based.

20

21 16. The method of Claim 1, wherein the device identifier
22 and the private key are rewritable at one time, but as of
23 the time step (a) occurs are no longer rewritable.

24

25 17. The method of Claim 1, further comprising:
26 receiving on a license manager the device identifier
27 maintained on the programmable logic device;
28 receiving on the license manager a first authorization
29 code; and
30 determining whether the first authorization code has a
31 predetermined relationship with respect to the device
32 identifier, wherein if the first authorization code is

1 determined to have the predetermined relationship then
2 the license manager sends the first encrypted key to the
3 programmable logic device such that it is received in
4 step (b), and wherein if the first authorization code is
5 determined not to have the predetermined relationship
6 then the license manager does not send the first
7 encrypted key to the programmable logic device in step
8 (b)

9
10 18. The method of Claim 17, wherein the first
11 authorization code has the predetermined relationship
12 with respect to the device identifier if the first
13 authorization code contains the device identifier in an
14 encrypted form.

15
16 19. A method comprising:

17 receiving onto a programmable logic device an
18 encrypted first key;

19 on the programmable logic device decrypting the
20 encrypted first key to generate a first key and storing
21 the first key on the programmable logic device;

22 receiving onto the programmable logic device a
23 configuration bitstream having a first portion and a
24 second portion;

25 on the programmable logic device decrypting the first
26 portion of the configuration bitstream using the first
27 key;

28 configuring the programmable logic device with the
29 decrypted first portion of the configuration bitstream
30 thereby realizing a first IP module;.
31

1 20. A programmable logic device that receives an encrypted
2 configuration bitstream, the programmable logic device
3 comprising:

4 non-volatile storage that stores a first key;
5 a decryptor that decrypts a first part of the
6 encrypted configuration bitstream using the first key and
7 thereby generates first configuration data; and
8 first configurable logic elements being configured by
9 the first configuration data.

10
11 21. The programmable logic device of Claim 20, wherein the
12 first part of the encrypted configuration bitstream is
13 identified by a first key number in the bitstream, the
14 first key number identifying the first key in the non-
15 volatile storage.

16
17 22. A method, comprising:

18 receiving on a development system a device identifier
19 from a programmable logic device;

20 receiving on the development system an authorization
21 code;

22 verifying on the development system that the
23 authorization code and the device identifier have a
24 predetermined relationship, wherein if the authorization
25 code and the device identifier have the predetermined
26 relationship then encrypting a key using the device
27 identifier and sending the encrypted key from the
28 development system to the programmable logic device, but
29 wherein if the authorization code and the device
30 identifier do not have the predetermined relationship
31 then the encrypted key is not sent from the development
32 system to the programmable logic device; and

1 the development system using the key to encrypt a
2 portion of a configuration data bitstream, the
3 development system outputting the configuration data
4 bitstream including the encrypted portion.

5
6 23. The method of Claim 22, wherein the key has a key
7 number, and wherein the development system adds the key
8 number to the configuration data bitstream such that the
9 key number is associated with the encrypted portion of
10 the configuration data bitstream, the configuration data
11 bitstream output from the development system including
12 the encrypted portion and the key number.

13
14 24. The method of Claim 22, wherein the development system
15 comprises a capture/design tool and a license manager,
16 the method further comprising:

17 if the authorization code and the device identifier
18 are verified as having the predetermined relationship
19 then the license manager allows use of IP module design
20 information by the capture/design tool, whereas if the
21 authorization code and the device identifier are not
22 verified as having the predetermined relationship then
23 the license manager does not allow use of the IP module
24 design information by the capture/design tool.

25
26 25. The method of Claim 22, wherein the portion of the
27 configuration data bitstream is configuration data for an
28 IP module, the development system comprising a
29 capture/design tool, the capture/design tool being usable
30 to view a net external to the IP module, the
31 capture/design tool not being usable to view a net
32 internal to the IP module.

1
2 26. A development system, comprising:

3 a capture/design tool; and

4 means for verifying that an authorization code has a
5 predetermined relationship with respect to a device
6 identifier read from a programmable logic device, and if
7 the authorization code is verified then the means also
8 encrypting a key and sending the encrypted key to the
9 programmable logic device, if the authorization code is
10 verified then the means also uses the key to encrypt a
11 portion of a configuration data bitstream output by the
12 capture/design tool, the configuration data bitstream
13 including the encrypted portion being sent to the
14 programmable logic device.

15
16 27. The development system of Claim 26, wherein the
17 encrypted portion of the bitstream contains configuration
18 data for an IP module, the capture/design tool being
19 usable to view a net external to the IP module, the
20 capture/design tool being unusable to view a net internal
21 to the IP module.

22
23 28. The development system of Claim 26, wherein the key
24 has a key number, the means inserting the key number into
25 the configuration data bitstream sent to the programmable
26 logic device, the key number in the configuration data
27 bitstream being associated with the encrypted portion of
28 the configuration data bitstream.

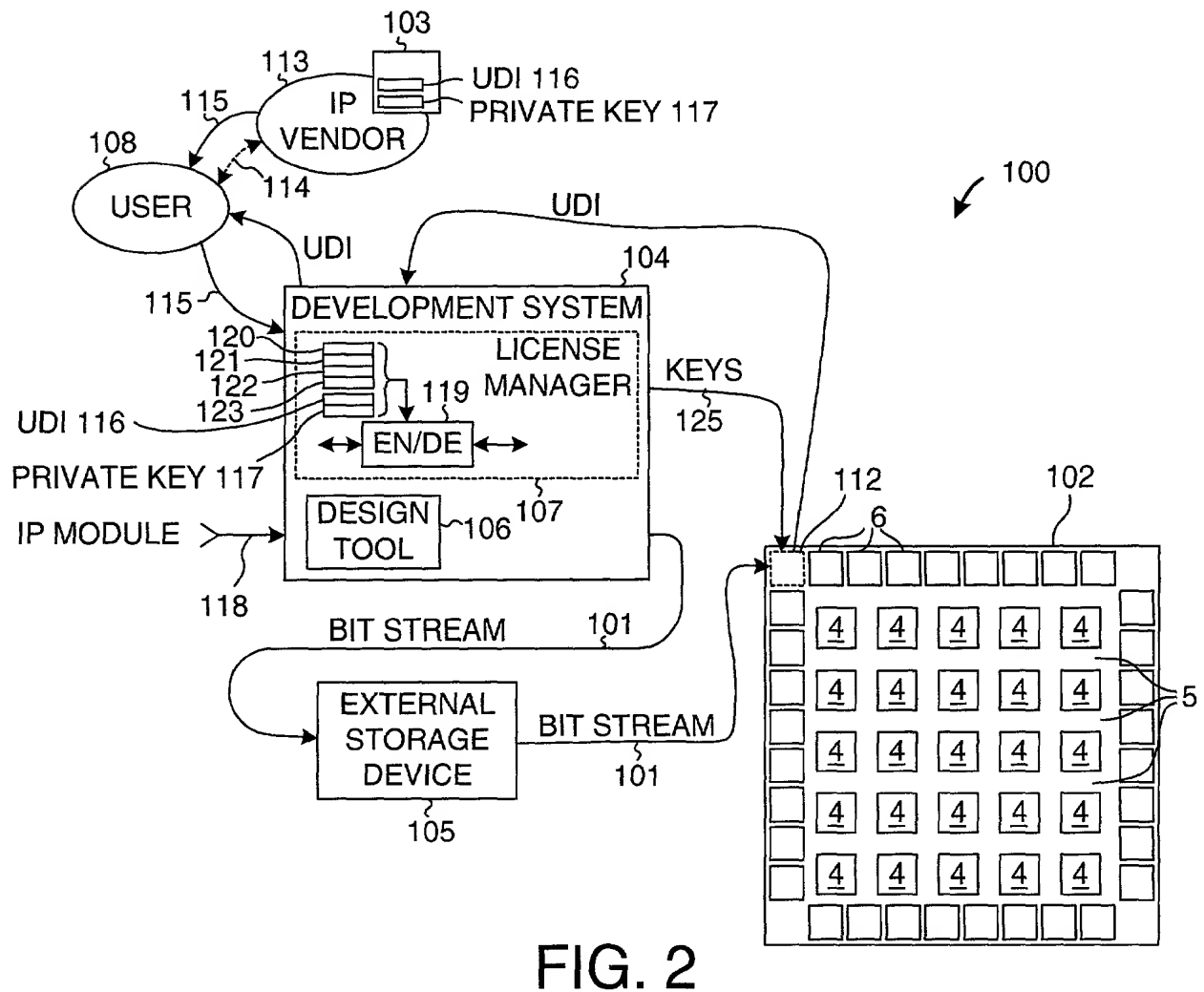
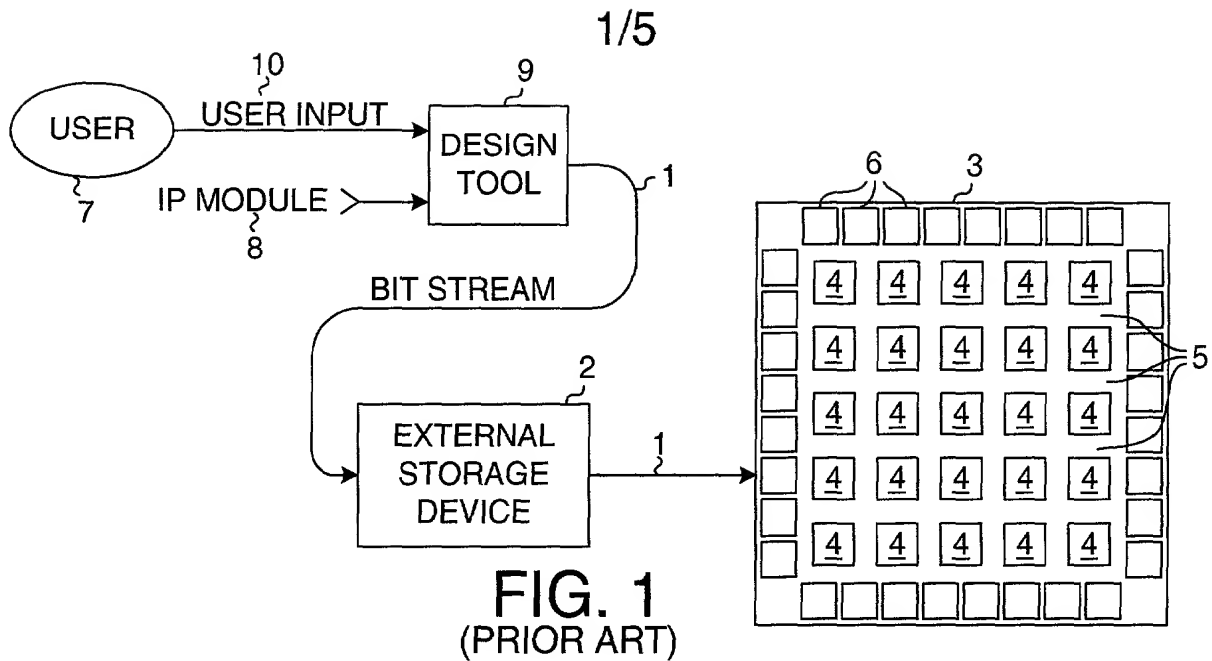
INTELLECTUAL PROPERTY PROTECTION
IN A PROGRAMMABLE LOGIC DEVICE

David B. Parlour

Richard S. Ballantyne

ABSTRACT OF THE DISCLOSURE

Individual IP vendors can directly license their IP modules to PLD users under their own chosen conditions. Individual IP vendors can use time-based licenses or use-based licenses. Each PLD is manufactured with a non-volatile non-rewritable unique device identifier (UDI) that uniquely identifies the PLD. If, for example, a user obtains a license to use a particular IP module on a particular PLD, then the IP vendor issues the user an authorization code that contains the UDI (in encrypted form) of the authorized target PLD. When the user wishes to use the IP module, the user supplies the authorization code to a license manager. The license manager decrypts the authorization code and checks that the UDI of the supplied authorization code matches the UDI of the to-be-programmed target PLD. If the two match, then the license manager encrypts a key for the IP module using the UDI and a private key, and then sends the encrypted key to the target PLD. The target PLD uses its UDI and the private key to decrypt the key. The key is then stored on the PLD. When the configuration bitstream for the design that incorporates the IP module is later sent to the PLD, the license manager encrypts the IP module portion with the key. The PLD receives the bitstream and uses the stored key to decrypt the IP module portion of the bitstream. The unencrypted bitstream is then used to configure the PLD.



2/5

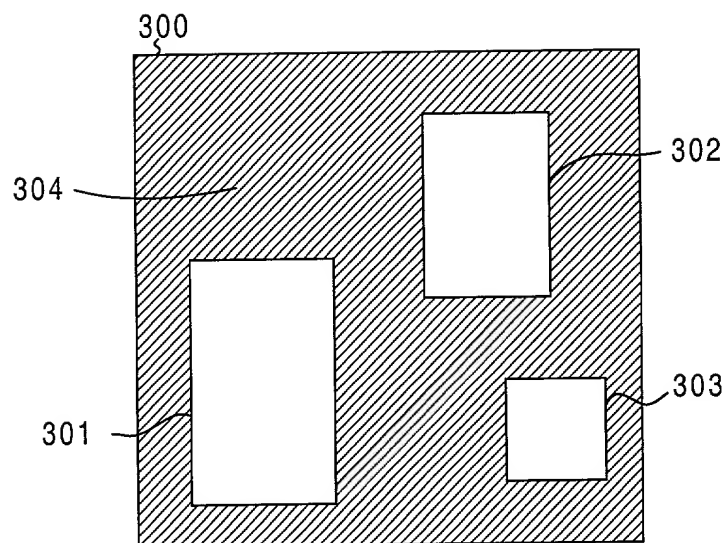


FIG. 3

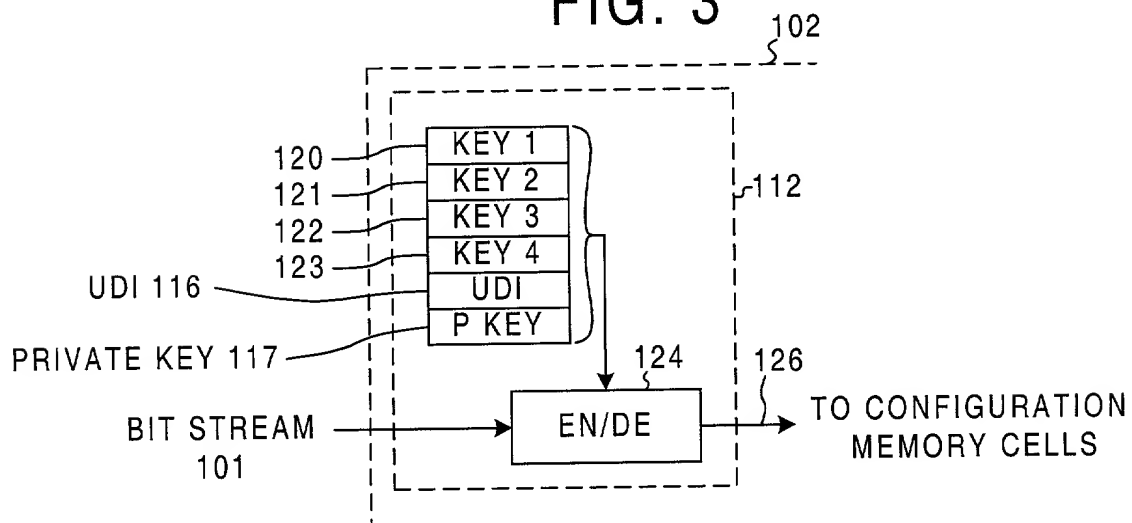


FIG. 5

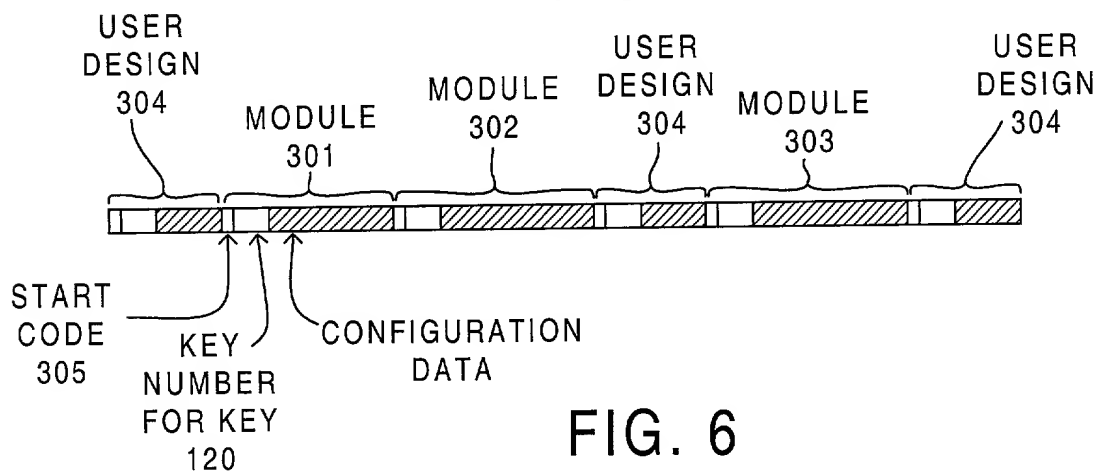
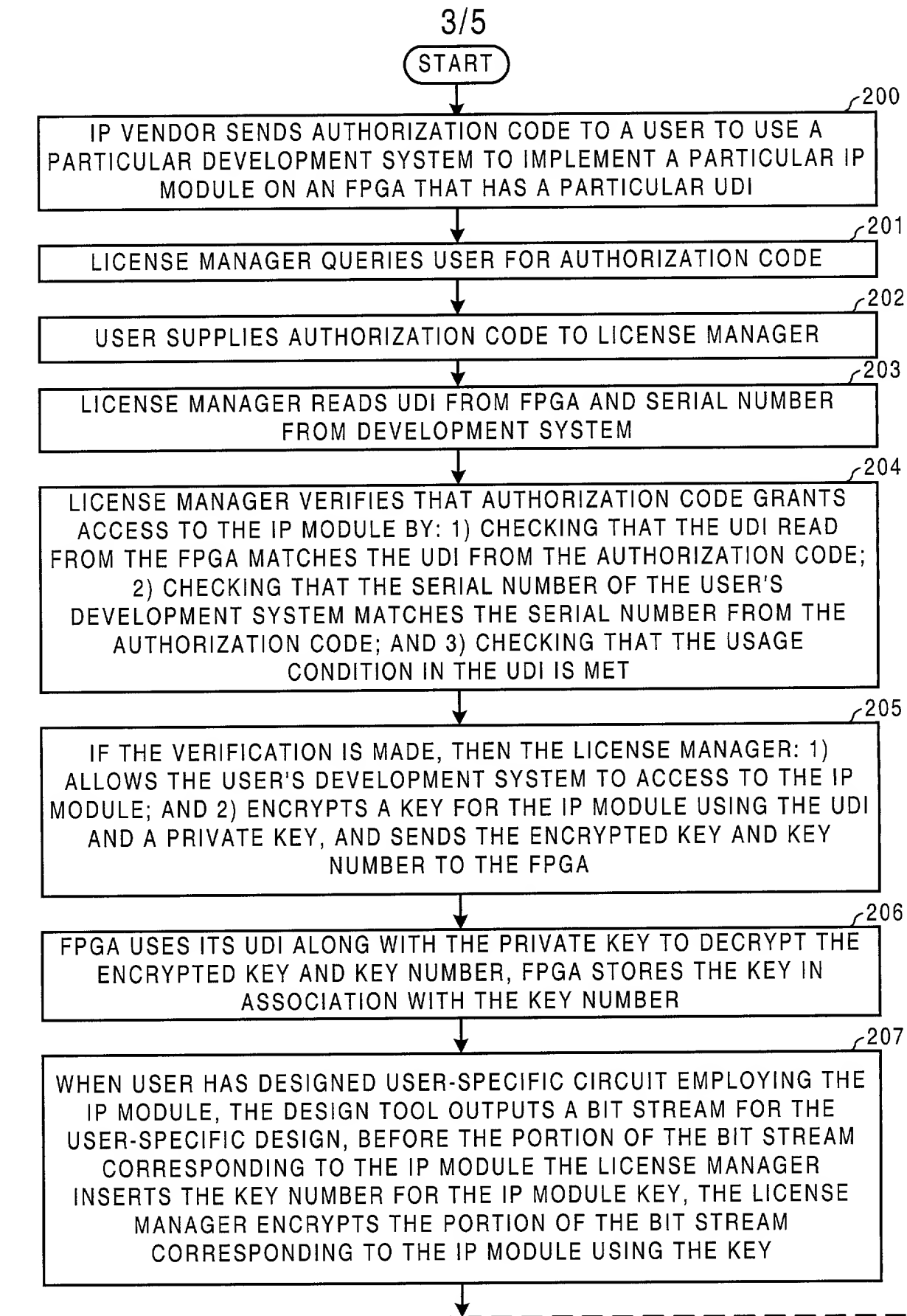


FIG. 6



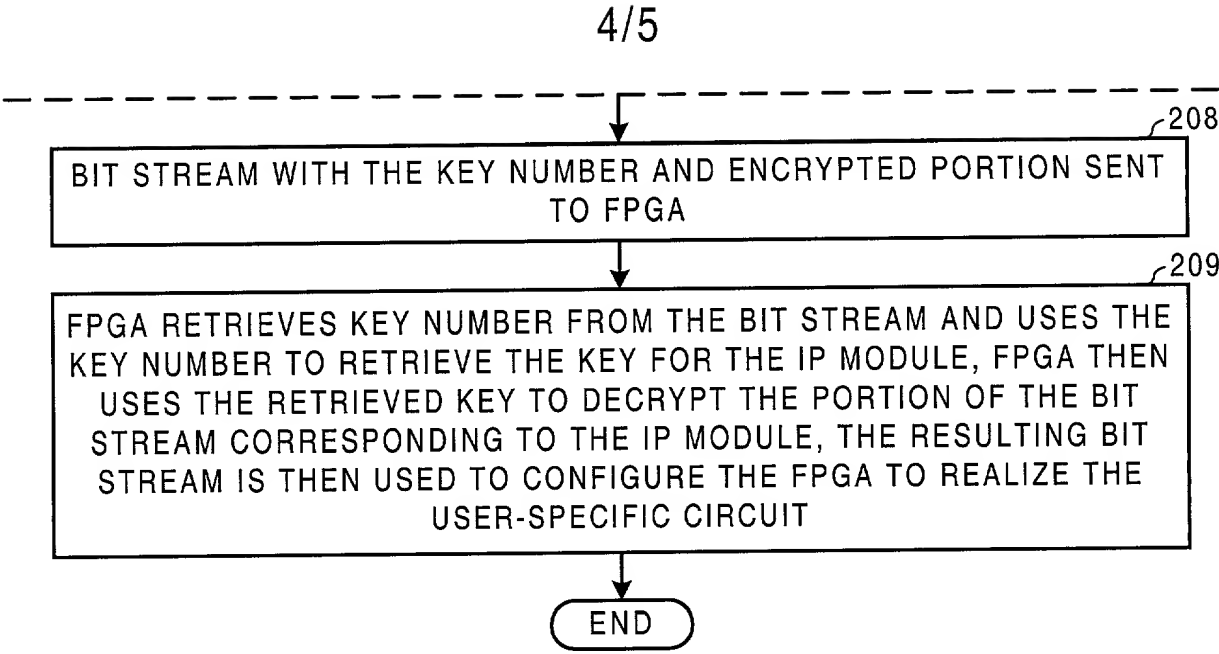


FIG. 4A

KEY TO FIG. 4

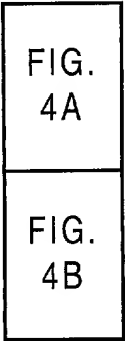


FIG. 4B

FIG. 4

5/5

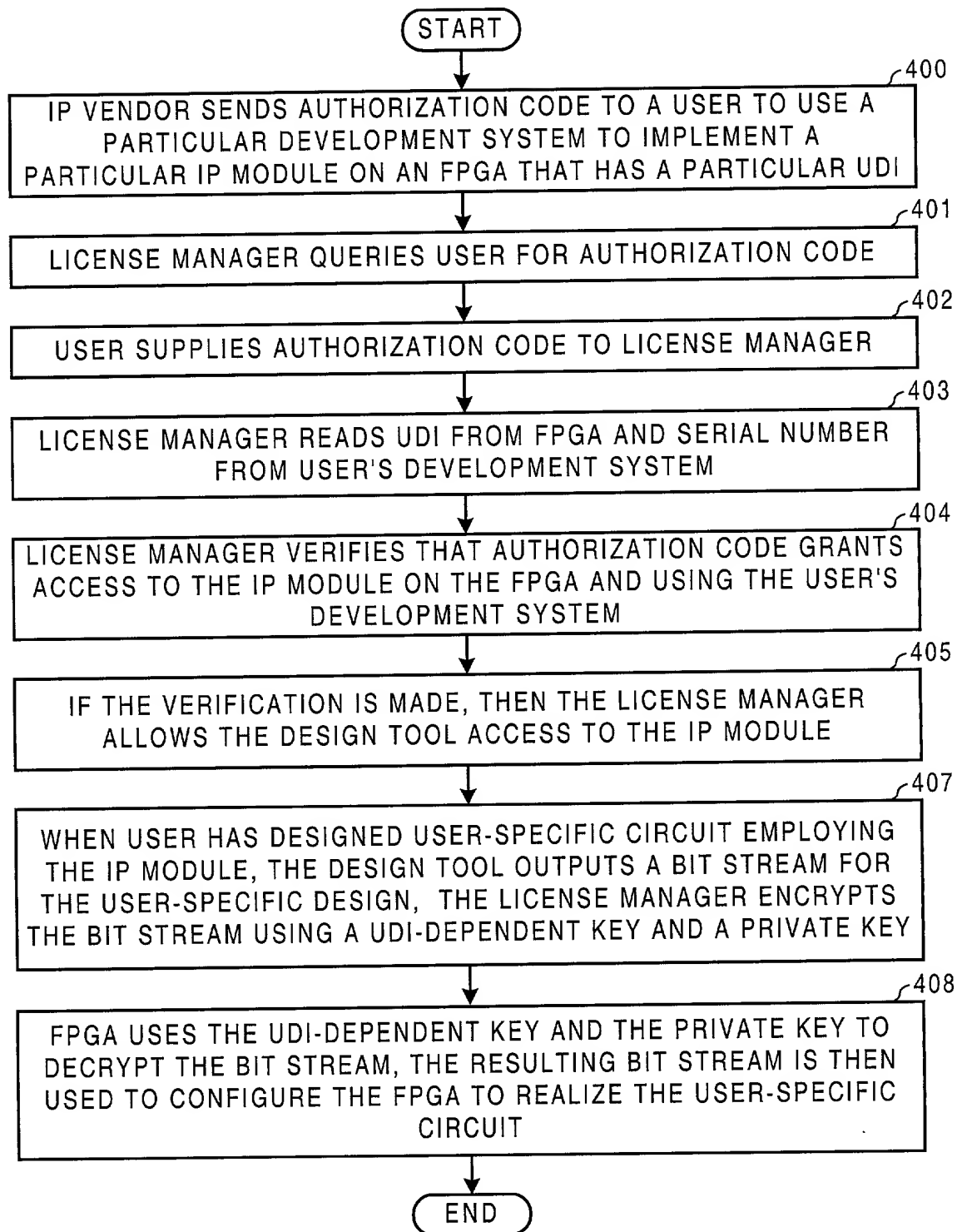


FIG. 7

DECLARATION FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below adjacent to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of subject matter (process, machine, manufacture, or composition of matter, or an improvement thereof) that is disclosed and/or claimed and for which a patent is solicited by way of the application entitled

INTELLECTUAL PROPERTY PROTECTION IN A PROGRAMMABLE LOGIC DEVICE

which (check)

- ☒ is attached hereto.
☐ and is amended by the Preliminary Amendment attached hereto.
☐ was filed on _____ as Application Serial No. _____.
☐ and was amended on _____ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified application, including the claims, including portions amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, § 1.56(a).

I hereby claim foreign priority benefits under title 35, United States Code, § 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
			Yes	No
_____ (Number)	_____ (Country)	_____ (Day/Month/Year Filed)		
_____ (Number)	_____ (Country)	_____ (Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under 35 U.S.C. § 119(e) of any United States provisional application(s) listed below

(Application Number(s))	(Filing Date (MM/DD/YYYY))
_____	_____

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s) listed below and, insofar as any subject matter of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, § 1.56(a) which occurred between the filing date of the prior application(s) and the national or PCT international filing date of this application:

(Application Serial No.)	(Filing Date)	(Status-patented, pending, abandoned)
_____	_____	_____
_____	_____	_____

I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and to transact all business in the United States Patent and Trademark Office connected herewith:

Edel M. Young (32,451), and Lois D. Cartier (40,941) located at Xilinx, Inc., 2100 Logic Drive, San Jose, California 95124, and
Arthur J. Behiel (39,603) located at the Law Office of Arthur Joseph Behiel, 7041 Koll Center Parkway, Suite 280, Pleasanton, California 94566.

Address all telephone calls to: Edel M. Young at Telephone No. (408) 879- 4969
Address all correspondence to: Edel M. Young
Xilinx, Inc.
2100 Logic Drive
San Jose, California 95124

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Title 18, United States Code, § 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of sole or first inventor David B. Parlour

Inventor's signature _____ Date _____

Residence 4231 Parkman Avenue, Pittsburg, PA 15213 Citizenship Canada

Post Office Address 4231 Parkman Avenue, Pittsburg, PA 15213

Full name second inventor Richard S. Ballantyne

Inventor's signature  Date March 3/2011

Residence 34 Hampel Crescent, Stittsville, Ontario K2S1E4 Citizenship Canada

Post Office Address 34 Hampel Crescent, Stittsville, Ontario K2S1E4 Canada

Full name third inventor _____

Inventor's signature _____ Date _____

Residence _____ Citizenship _____

Post Office Address _____

Full name fourth inventor _____

Inventor's signature _____ Date _____

Residence _____ Citizenship _____

Post Office Address _____

DECLARATION FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below adjacent to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of subject matter (process, machine, manufacture, or composition of matter, or an improvement thereof) that is disclosed and/or claimed and for which a patent is solicited by way of the application entitled

INTELLECTUAL PROPERTY PROTECTION IN A PROGRAMMABLE LOGIC DEVICE

which (check)

- ☒ [X] is attached hereto.
☐ [] and is amended by the Preliminary Amendment attached hereto.
☐ [] was filed on _____ as Application Serial No. _____.
☐ [] and was amended on _____ (if applicable).

I hereby state that I have reviewed and understand the contents of the above identified application, including the claims, including portions amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, § 1.56(a).

I hereby claim foreign priority benefits under title 35, United States Code, § 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
			Yes	No
_____	_____	_____		
(Number)	(Country)	(Day/Month/Year Filed)		
_____	_____	_____		
(Number)	(Country)	(Day/Month/Year Filed)		

I hereby claim the benefit under 35 U.S.C. § 119(e) of any United States provisional application(s) listed below

_____	_____
(Application Number(s))	(Filing Date (MM/DD/YYYY))

I hereby claim the benefit under Title 35, United States Code, § 120 of any United States application(s) listed below and, insofar as any subject matter of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, § 1.56(a) which occurred between the filing date of the prior application(s) and the national or PCT international filing date of this application:

_____	_____	_____
(Application Serial No.)	(Filing Date)	(Status-patented, pending, abandoned)
_____	_____	_____
(Application Serial No.)	(Filing Date)	(Status-patented, pending, abandoned)

I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and to transact all business in the United States Patent and Trademark Office connected herewith:

Edel M. Young (32,451), and Lois D. Cartier (40,941) located at Xilinx, Inc., 2100 Logic Drive, San Jose, California 95124, and

Arthur J. Behiel (39,603) located at the Law Office of Arthur Joseph Behiel, 7041 Koll Center Parkway, Suite 280, Pleasanton, California 94566.

Address all telephone calls to: Edel M. Young at Telephone No. (408) 879- 4969

Address all correspondence to: Edel M. Young
Xilinx, Inc.
2100 Logic Drive
San Jose, California 95124

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Title 18, United States Code, § 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of sole or first inventor David B. Parlour

Inventor's signature David B Parlour Date 10th March, 2000

Residence 4231 Parkman Avenue, Pittsburg, PA 15213 Citizenship Canada

Post Office Address 4231 Parkman Avenue, Pittsburg, PA 15213

Full name second inventor Richard S. Ballantyne

Inventor's signature _____ Date _____

Residence 34 Hampel Crescent, Stittsville, Ontario K2S1E4 Citizenship Canada

Post Office Address 34 Hampel Crescent, Stittsville, Ontario K2S1E4 Canada

Full name third inventor _____

Inventor's signature _____ Date _____

Residence _____ Citizenship _____

Post Office Address _____

Full name fourth inventor _____

Inventor's signature _____ Date _____

Residence _____ Citizenship _____

Post Office Address _____